



NAVAL
POSTGRADUATE
SCHOOL

An information-theoretic approach to software test-retest problems

May 13, 2010

Karl D. Pfeiffer

Valery A. Kanevsky

Thomas J. Housel

Monterey, California

WWW.NPS.EDU



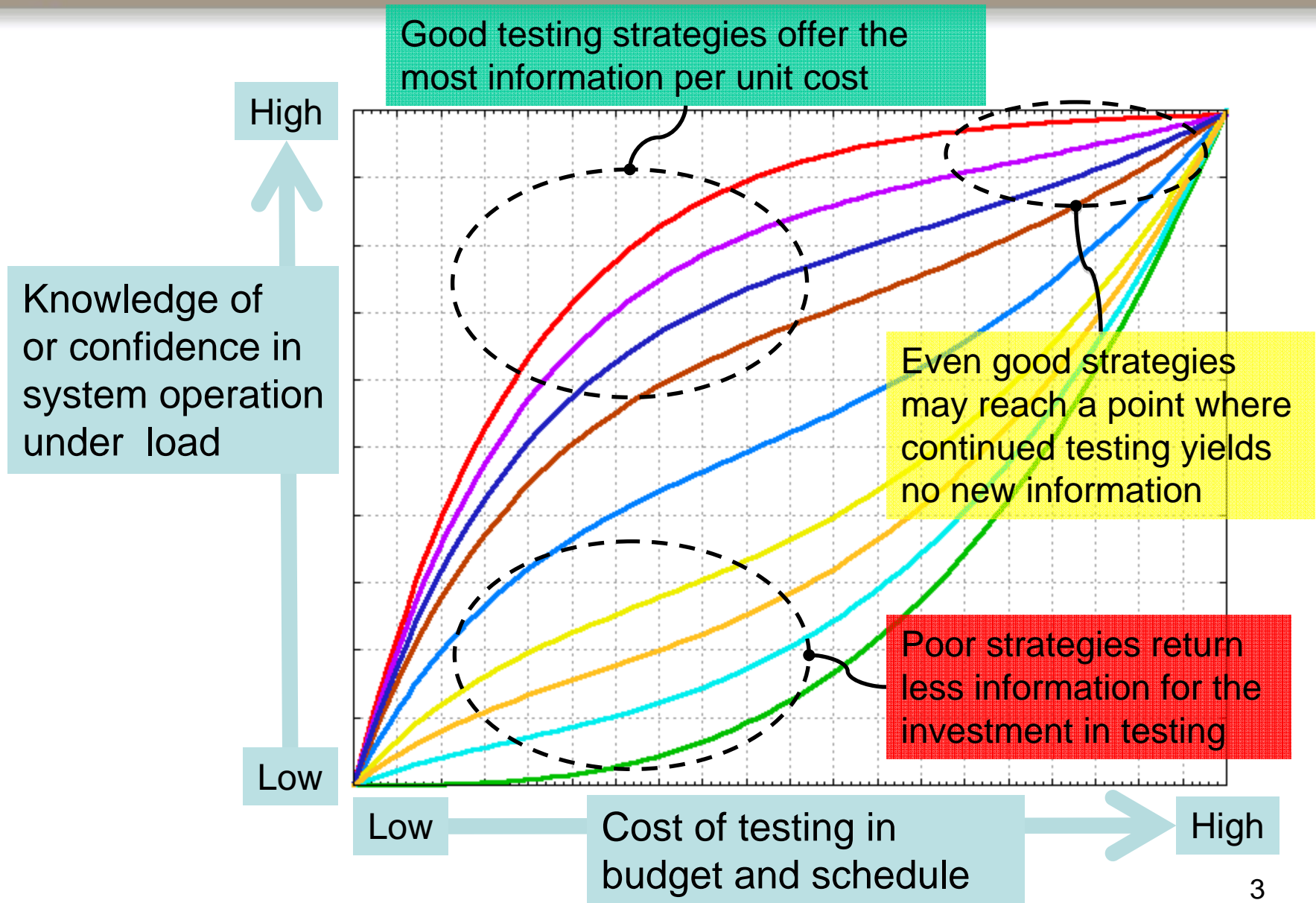
Report Documentation Page				Form Approved OMB No. 0704-0188	
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
1. REPORT DATE 13 MAY 2010		2. REPORT TYPE		3. DATES COVERED 00-00-2010 to 00-00-2010	
4. TITLE AND SUBTITLE An information-theoretic approach to software test-retest problems				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School, Monterey, CA, 93943				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited					
13. SUPPLEMENTARY NOTES 7th Annual Acquisition Research Symposium to be held May 12-13, 2010 in Monterey, California.					
14. ABSTRACT					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT Same as Report (SAR)	18. NUMBER OF PAGES 21	19a. NAME OF RESPONSIBLE PERSON
a. REPORT unclassified	b. ABSTRACT unclassified	c. THIS PAGE unclassified			



- Open architectures (OA) and reusable software components offer the promise of more rapid fielding of increments in systems development
- Testing and re-testing these components requires a significant level of effort as new systems are developed and old systems are upgraded
- *How much testing is enough? When can we stop testing?*



Motivation





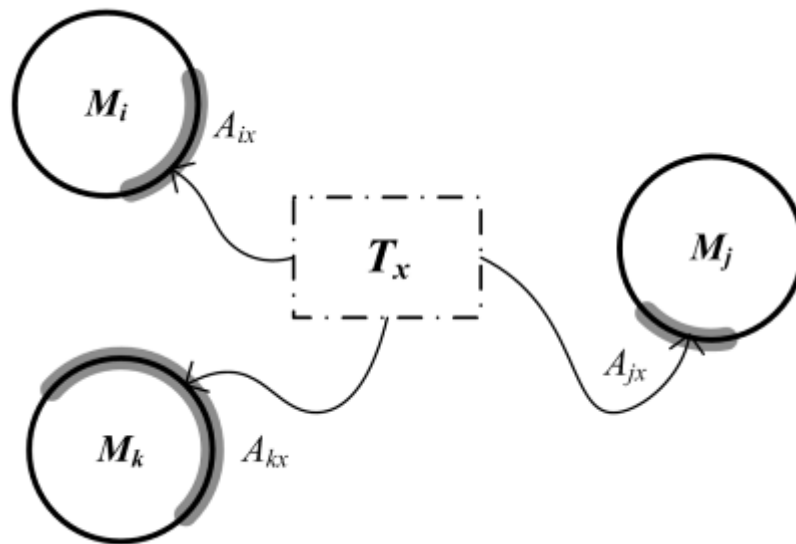
- We can identify good testing strategies by constructing a simple model of the system, its components, and its attendant test suite
- This model requires
 - Estimates of a prior probability of failure for modules within the system
 - Estimates of the coverage for each test in the suite over these modules
- These estimates need not be precise to make the model useful
 - Monte Carlo simulation can be used to sample around the estimates as means, offering some insight into model sensitivity



- This model should help answer questions like:
 - Given a desired level of confidence in system operation, how much testing should we accomplish? How much will this cost?
 - Given a fixed budget for testing, how much confidence in system performance can we achieve through testing?
 - Given a particular test suite, how much information is attainable given infinite resources?



Model fundamentals



- A module M_i is modeled as a unit circle with probability of being defective b_i
 - Test T_x exercises region A_{ix} in module M_i
 - In general we assume that T_x may exercise several regions across several modules
-
- A test has two possible outcomes:
 - *PASS* indicates that the test did not *detect* a defect in any of the exercised regions within the modules tested
 - *FAIL* indicates that at least one module exercised is defective, though we may not know which one



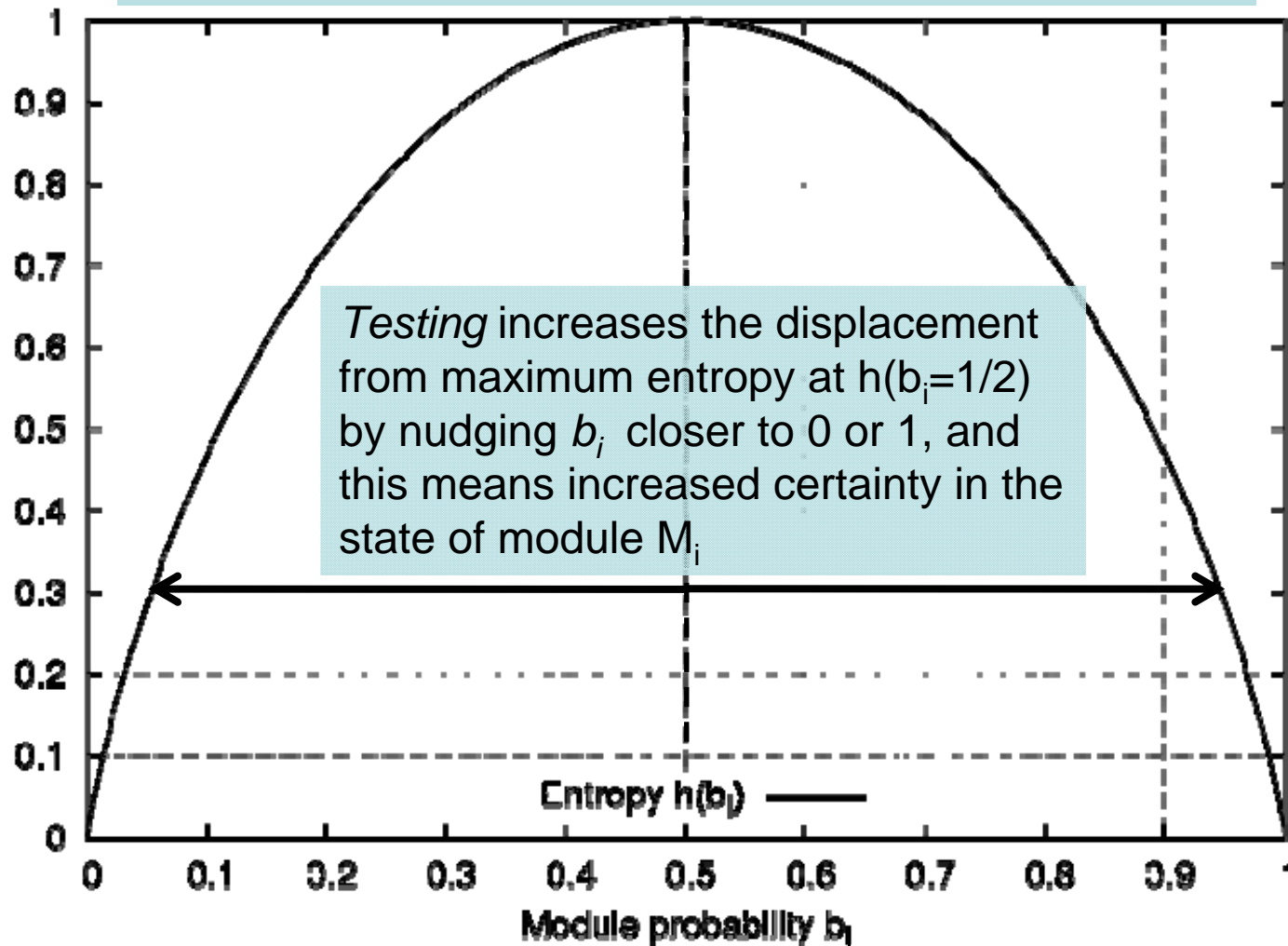
- These ambiguities offer a rich framework for modeling realistic system testing scenarios
 - We need not execute (and pay for) Tx to *forecast* information returned
 - Within this language of expression we can formulate a *quantitative* assessment of the information returned by a test sequence
- Across the system of modules M_i we can measure the information returned by a test using the classic residual entropy for a distribution of probabilities:

$$H = \sum_i h_i = \sum_i -b_i \log_2 b_i - (1 - b_i) \log_2 (1 - b_i)$$



Model fundamentals

At maximum entropy we have a 50/50 chance that our module is good or bad—we *might as well flip a coin*





- From entropy, we derive the forecast measure:

$$Q(T_x) = \sum_i \left(\max(b_i^{fail}, 1 - b_i^{fail}) P(T_x \text{ fails}) + \max(b_i^{pass}, 1 - b_i^{pass}) P(T_x \text{ passes}) \right)$$

- Let c_x be the cost of executing test T_x in appropriate units of time or money (or both) A *good* strategy will sequence the suite of tests such that:

$$\frac{Q(T_{[1]})}{c_{[1]}} \geq \frac{Q(T_{[2]})}{c_{[2]}} \dots \geq \frac{Q(T_{[m]})}{c_{[m]}}$$

- These ratios represent *information per unit cost*



Model implementation

- Within the decision aid, for simple investigations, a fully randomized system can be created with only a few user specified constraints
- If the user has a few system details but only vague insight about others, these aspects can be augmented with randomized parameters (e.g. sizes and number of coverages)
- A system with well-documented interdependencies can be completely specified by the user in terms of modules, tests and coverages

Animal

Dog





Model implementation

Risk-based Testing System Simulation

Execution Parameters

Case Name

Random Seed

Number of Trials

Defects per Trial

Reconfigure tests per trial?

Strategy

System Generation Parameters

Number of Modules

Number of Tests

	Min	Max
Modules per Test	<input type="text" value="1"/>	<input type="text" value="4"/>
Tests per Module	<input type="text" value="1"/>	<input type="text" value="4"/>
Coverage per Test	<input type="text" value="0.100"/>	<input type="text" value="1.000"/>
Failure rate	<input type="text" value="0.10000000"/>	<input type="text" value="0.30000000"/>

Main

Generate System

Load System Object

Load Configuration

Save Configuration

Execute Configuration

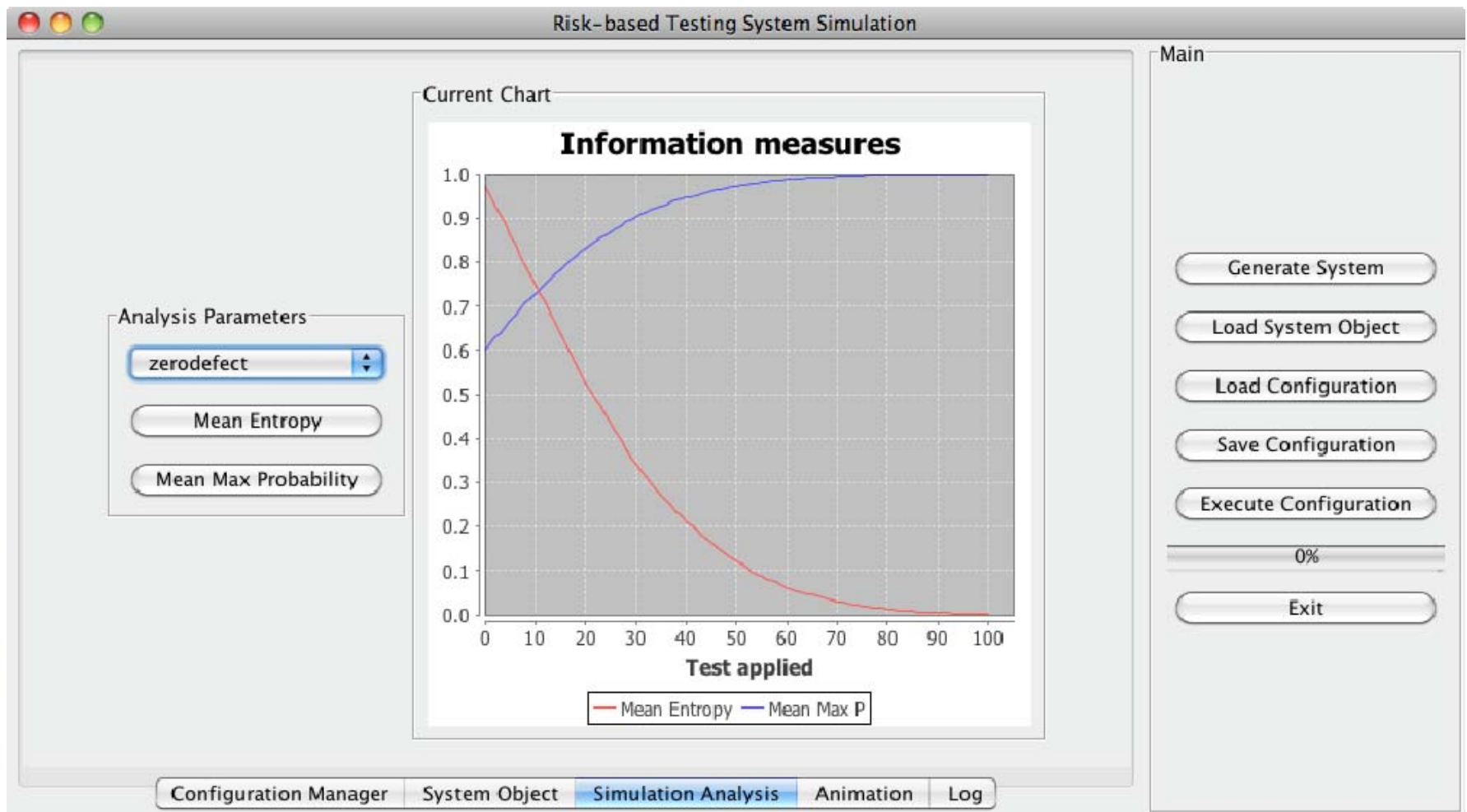
0%

Exit

Configuration Manager System Object Simulation Analysis Animation Log

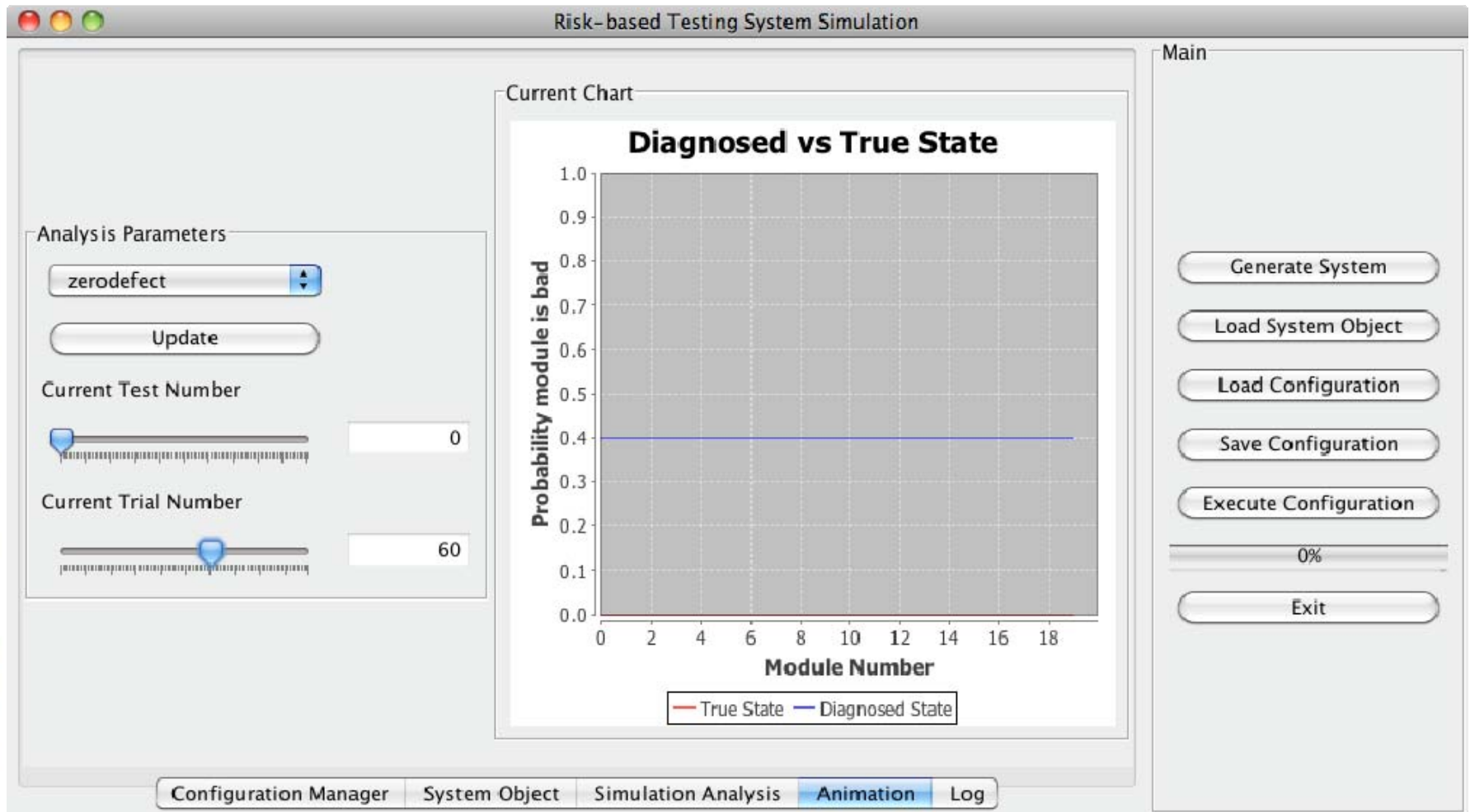


Model implementation



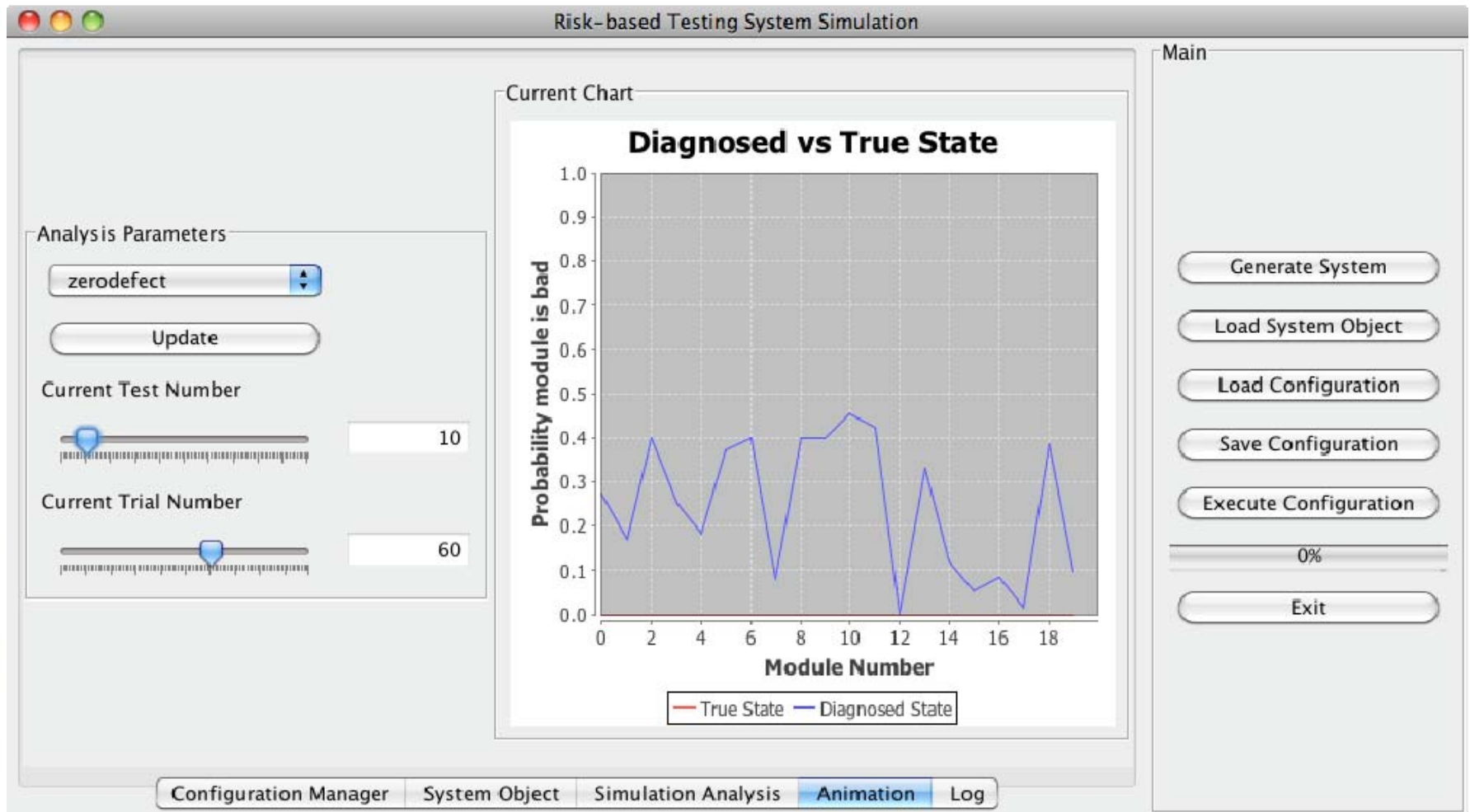


Model implementation



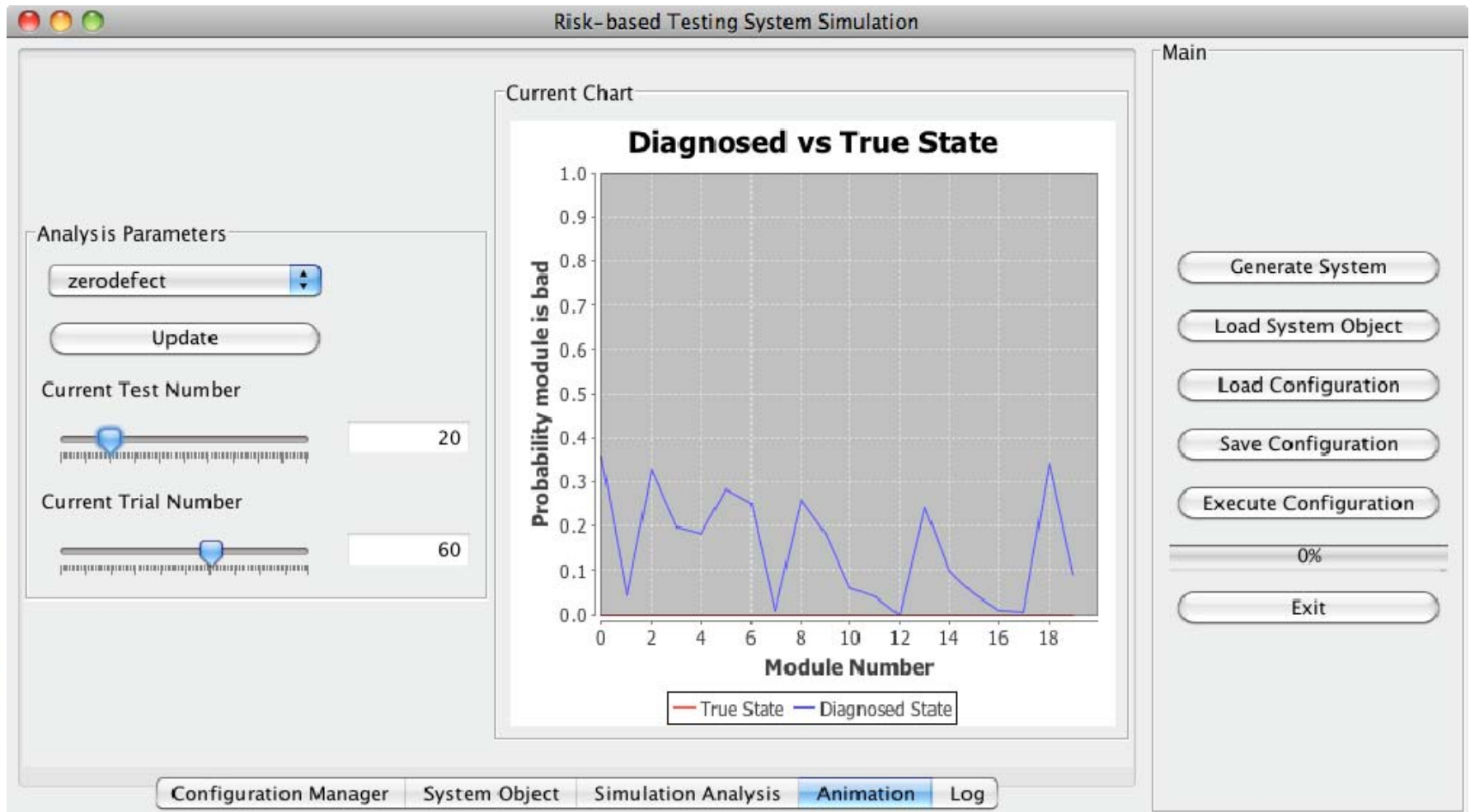


Model implementation



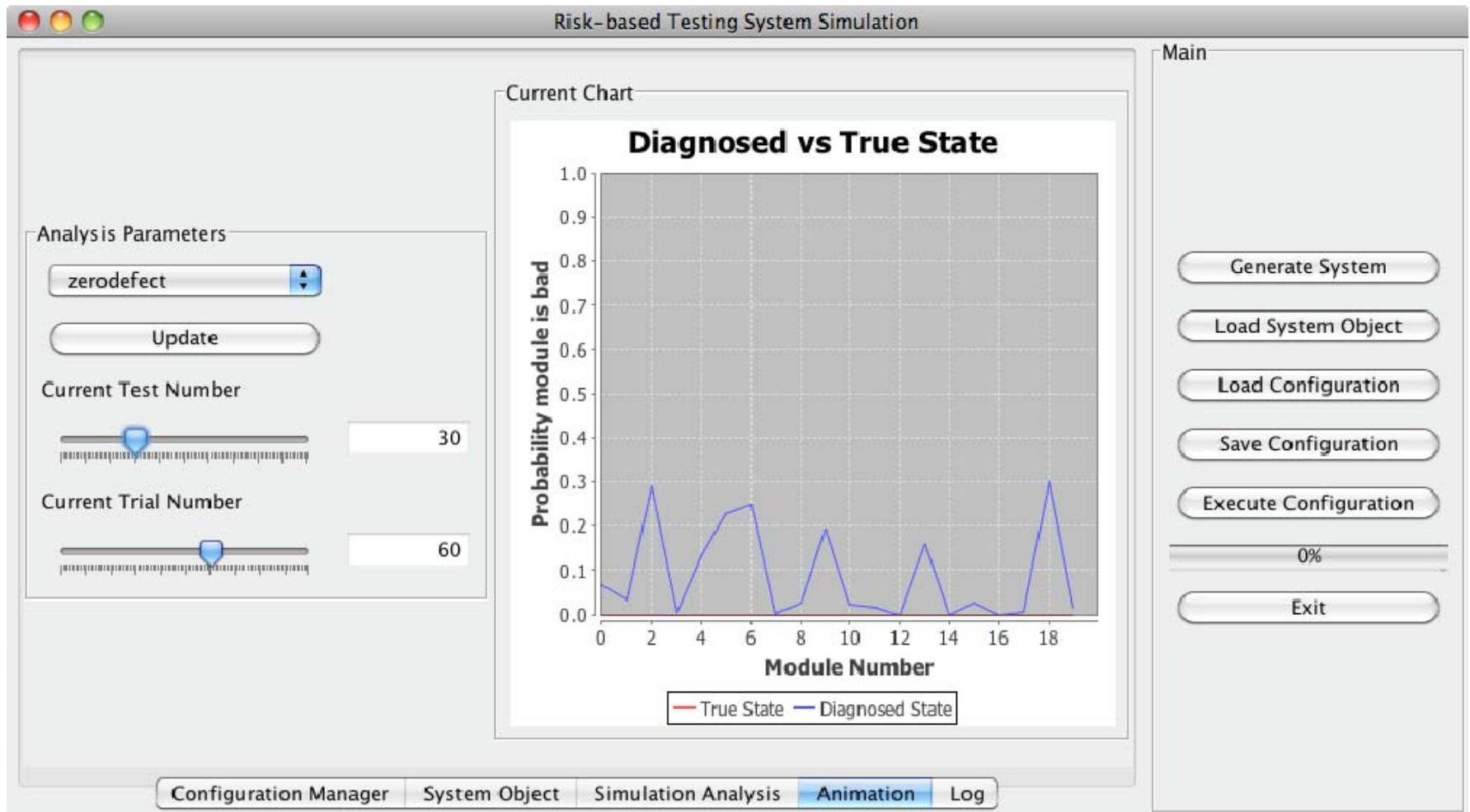


Model implementation



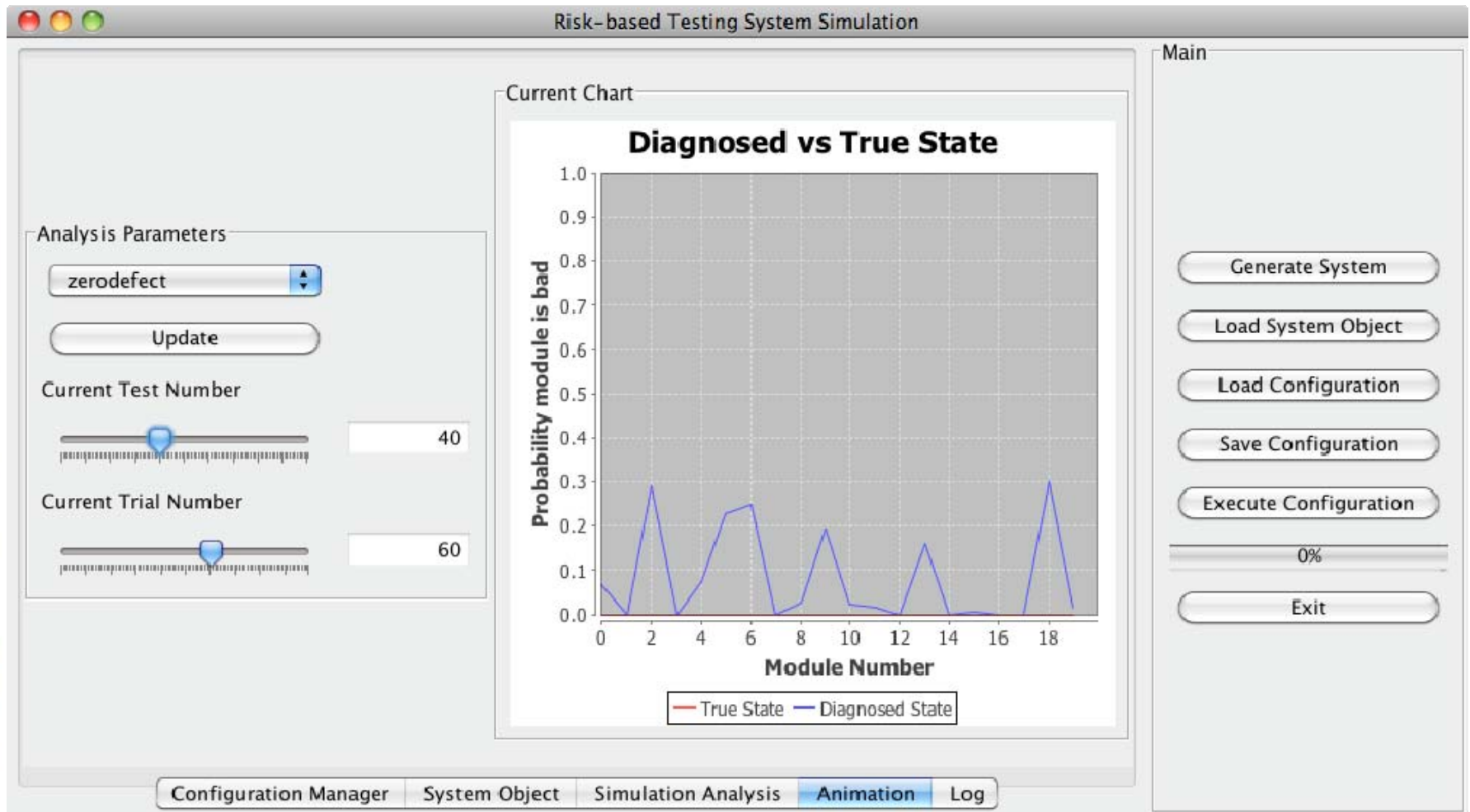


Model implementation



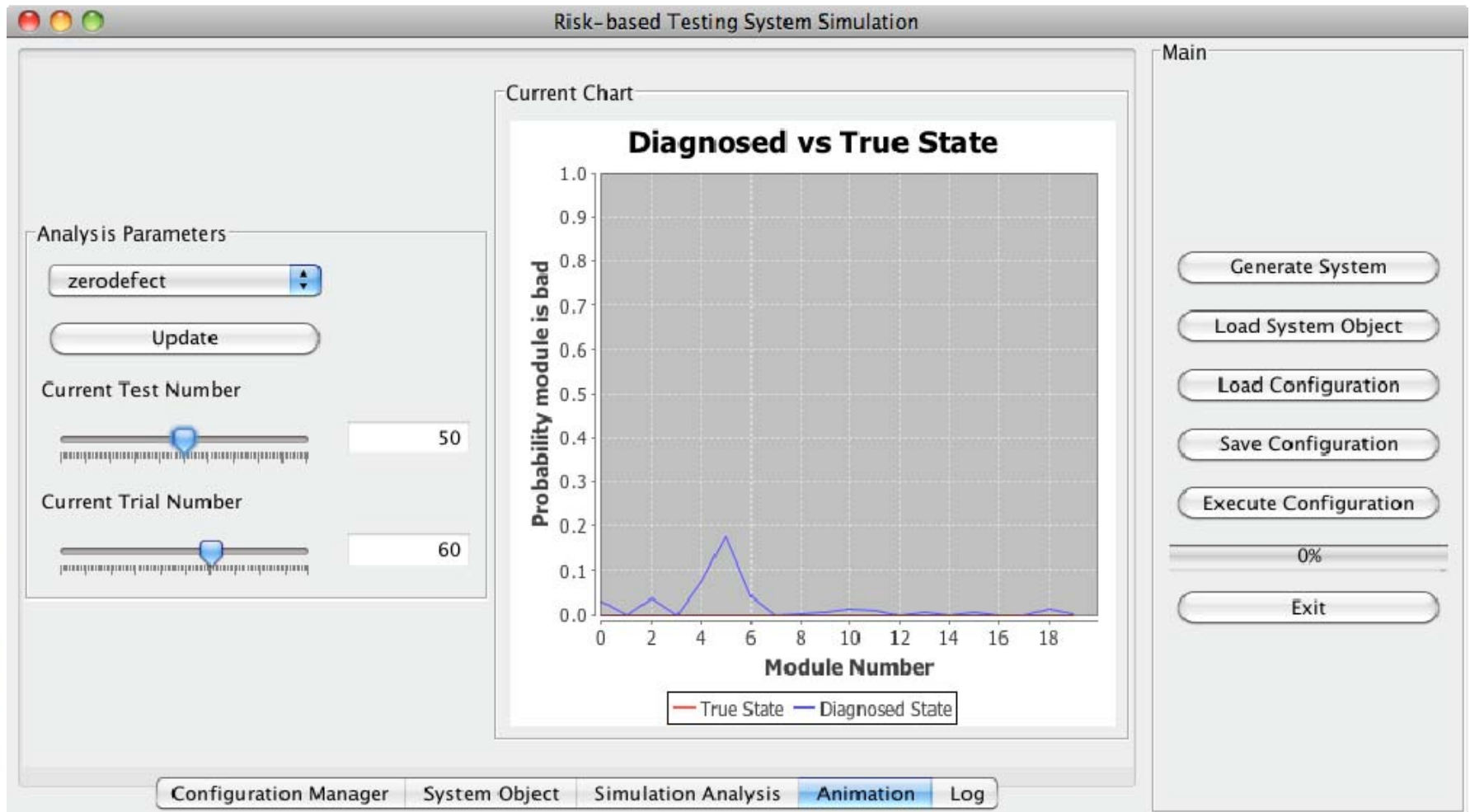


Model implementation



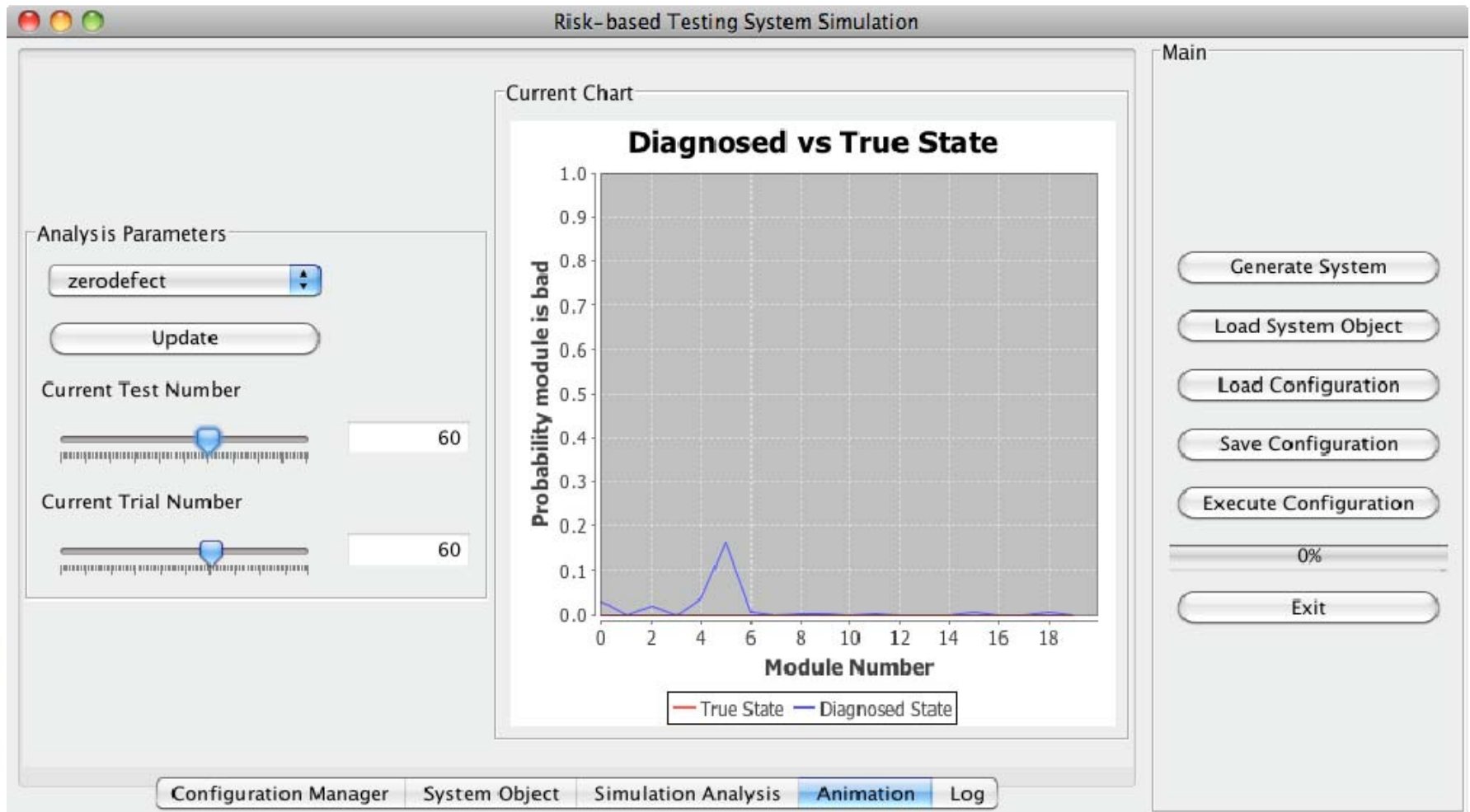


Model implementation



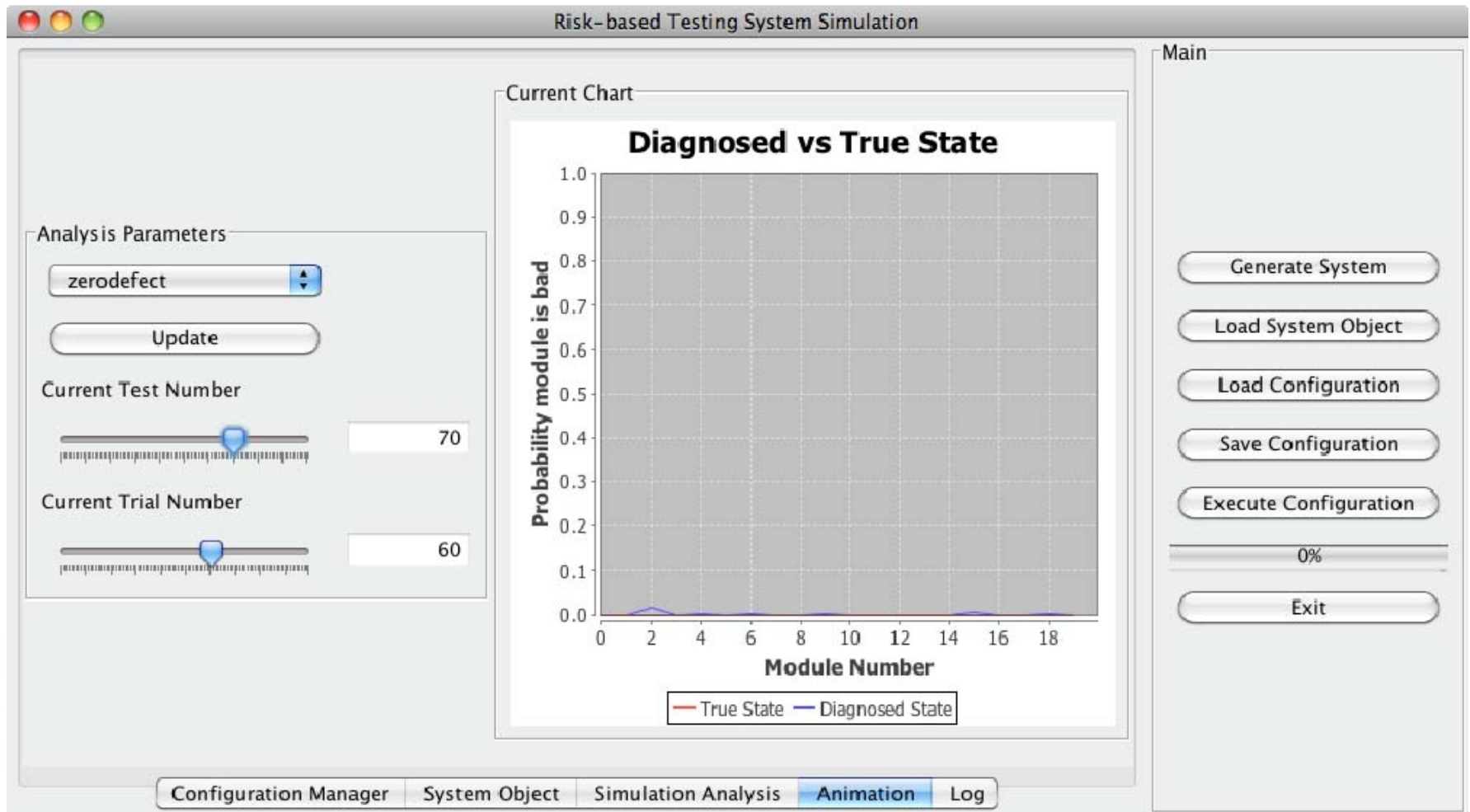


Model implementation





Model implementation





But, what does it all mean?

- *Effective, cost-efficient testing is critical to the long-term success of open architecture programs*
- This model and prototype decision aid provide a rigorous yet tractable way ahead to improve system testing
- Using this framework we can build the tools to:
 - Lower the testing costs for a given level of system reliability
 - Improve the use of existing suites for a given budget or schedule
 - Design better, more targeted test suites to minimize redundancy
 - Provide insight into the power or sensitivity of current test suites